

1 **CLAIMS**

2 What is claimed is:

3 1. A compiler device for optimizing a program which
4 manipulates a character string, the compiler device
5 comprising:

6 an append instruction detection unit for detecting an
7 append instruction to append a character string to a string
8 variable for storing a character string, in the program;

9 a store code generation unit for generating, as a
10 substitute for each of a plurality of the append
11 instructions detected by the append instruction detection
12 unit, a store code for storing data of an appendant
13 character string to be appended to the string variable by
14 the append instruction into a buffer, the plurality of
15 append instructions appending the character strings to the
16 same string variable; and

17 an append code generation unit for generating an append
18 code for appending a plurality of the appendant character
19 strings to the string variable, at a position to be executed
20 before an instruction to refer to the string variable in the
21 program.

1 2. The compiler device according to claim 1, further
2 comprising:
3 a reference instruction detection unit for detecting a
4 reference instruction which first refers to the string
5 variable after the character strings have been appended to
6 the string variable by the plurality of append instructions,
7 wherein the append code generation unit generates the append
8 code at a position to be executed after the store codes and
9 before the reference instruction.

10 3. The compiler device according to claim 1, wherein the
11 append instruction detection unit detects, as the append
12 instruction, a combination of:
13 an instruction to convert an immutable string variable
14 in which a process of appending a character string is not
15 allowed, into a mutable string variable in which a process
16 of appending a character string is allowed;
17 an instruction to append the appendant character string
18 to the mutable string variable; and
19 an instruction to convert the mutable string variable
20 into the immutable string variable.

1 4. A compiler device for optimizing a program which
2 manipulates a character string, the compiler device
3 comprising:

4 an append instruction detection unit for detecting an
5 append instruction to append a character string to a string
6 variable for storing a character string, in the program;

7 a store code generation unit for generating, as a
8 substitute for each of a plurality of the append
9 instructions detected by the append instruction detection
10 unit, a store code for storing an address in memory where an
11 appendant character string to be appended to the string
12 variable by the append instruction is stored, into a buffer,
13 the plurality of append instructions appending character
14 strings to the same string variable; and

15 an append code generation unit for generating an append
16 code for appending a plurality of the appendant character
17 strings stored in a plurality of the addresses, to the
18 string variable, at a position to be executed before an
19 instruction to refer to the string variable in the program.

20 5. A compiler device for optimizing a program which
21 manipulates a character string, the compiler device
22 comprising:

1 a mutable-to-immutable conversion instruction detection
2 unit for detecting a mutable-to-immutable conversion
3 instruction to convert a mutable string variable in which a
4 process of appending a character string is allowed, into an
5 immutable string variable in which a process of appending a
6 character string is not allowed;

7 an immutable-to-mutable conversion instruction
8 detection unit for detecting an immutable-to-mutable
9 conversion instruction to convert the immutable string
10 variable into the mutable string variable; and

11 an instruction elimination unit for eliminating the
12 immutable-to-mutable conversion instruction and for causing
13 the mutable string variable as a source variable of the
14 mutable-to-immutable conversion instruction, to be used as
15 the mutable string variable obtained from conversion by the
16 immutable-to-mutable conversion instruction after the
17 immutable-to-mutable conversion instruction, if an
18 instruction to be executed between the mutable-to-immutable
19 conversion instruction and the immutable-to-mutable
20 conversion instruction does not modify a character string
21 stored in the mutable string variable as the source variable
22 of the mutable-to-immutable conversion instruction, and if
23 an instruction to be executed between the

1 immutable-to-mutable conversion instruction and use of the
2 mutable string variable obtained from the conversion by the
3 immutable-to-mutable conversion instruction does not modify
4 any of the mutable string variable as the source variable of
5 the mutable-to-immutable conversion instruction and the
6 mutable string variable obtained from the conversion by the
7 immutable-to-mutable conversion instruction.

8 6. The compiler device according to claim 5, wherein the
9 instruction elimination unit further eliminates the
10 mutable-to-immutable conversion instruction if a character
11 string stored in the immutable string variable is not
12 referred to.

13 7. The compiler device according to claim 6, wherein the
14 instruction elimination unit moves the mutable-to-immutable
15 conversion instruction to each branch destination of a
16 branch instruction to be executed after the
17 mutable-to-immutable conversion instruction, and executes
18 partial dead assignment elimination for eliminating the
19 mutable-to-immutable conversion instruction if a character
20 string stored in the immutable string variable as a
21 destination variable of the mutable-to-immutable conversion

1 instruction is not referred to on each branch destination of
2 the branch instruction.

3 8. The compiler device according to claim 5, wherein the
4 immutable-to-mutable conversion instruction detection unit
5 detects, as the immutable-to-mutable conversion instruction,
6 a combination of:

7 an instruction to reserve a memory area to be used as a
8 mutable string variable; and

9 an instruction to append a character string stored in
10 the immutable string variable to the mutable string
11 variable.

12 9. The compiler device according to claim 5, further
13 comprising:

14 a partial redundancy elimination unit for executing a
15 partial redundancy elimination process of moving the
16 immutable-to-mutable conversion instruction detected by the
17 immutable-to-mutable conversion instruction detection unit
18 to each control flow edge which merges into a single control
19 flow before the immutable-to-mutable conversion instruction,

20

21 wherein the instruction elimination unit eliminates the

1 immutable-to-mutable conversion instruction, if an
2 instruction to be executed between the mutable-to-immutable
3 conversion instruction and the immutable-to-mutable
4 conversion instruction does not modify a character string
5 stored in the mutable string variable as the source variable
6 of the mutable-to-immutable conversion instruction and if an
7 instruction to be executed between the immutable-to-mutable
8 conversion instruction and the use of the mutable string
9 variable obtained from the conversion by the
10 immutable-to-mutable conversion instruction does not modify
11 any of the mutable string variable as the source variable of
12 the mutable-to-immutable conversion instruction and the
13 mutable string variable obtained from the conversion by the
14 immutable-to-mutable conversion instruction, in a program
15 obtained after the partial redundancy elimination process
16 has been executed.

17 10. The compiler device according to claim 9,
18 wherein the instruction elimination unit moves the
19 mutable-to-immutable conversion instruction to each branch
20 destination of a branch instruction to be executed after the
21 mutable-to-immutable conversion instruction, and executes
22 partial dead assignment elimination for eliminating the
23 mutable-to-immutable conversion instruction if a character

1 string stored in the immutable string variable as a
2 destination variable of the mutable-to-immutable conversion
3 instruction is not referred to on each branch destination of
4 the branch instruction.

5 11. A compiler program for optimizing a program which
6 manipulates a character string, by using a computer, the
7 compiler program causing the computer to function as:

8 an append instruction detection unit for detecting an
9 append instruction to append a character string to a string
10 variable for storing a character string, in the program;
11 a store code generation unit for generating, as a
12 substitute for each of a plurality of the append
13 instructions detected by the append instruction detection
14 unit, a store code for storing data of an appendant
15 character string to be appended to the string variable by
16 the append instruction into a buffer, the plurality of
17 append instructions appending the character strings to the
18 same string variable; and

19 an append code generation unit for generating an append
20 code for appending a plurality of the appendant character
21 strings to the string variable, at a position to be executed
22 before an instruction to refer to the string variable in the

1 program.

2 12. A compiler program for optimizing a program which
3 manipulates a character string, by using a computer, the
4 compiler program causing the computer to function as:

5 an append instruction detection unit for detecting an
6 append instruction to append a character string to a string
7 variable for storing a character string, in the program;

8 a store code generation unit for generating, as a
9 substitute for each of a plurality of the append
10 instructions detected by the append instruction detection
11 unit, a store code for storing an address in memory where an
12 appendant character string to be appended to the string
13 variable by the append instruction is stored, into a buffer,
14 the plurality of append instructions appending the character
15 strings to the same string variable; and

16 an append code generation unit for generating an append
17 code for appending a plurality of the appendant character
18 strings stored in a plurality of the addresses, to the
19 string variable, at a position to be executed before an
20 instruction to refer to the string variable in the program.

21 13. A compiler program for optimizing a program which

1 manipulates a character string, by using a computer, the
2 compiler program causing the computer to function as:
3 a mutable-to-immutable conversion instruction detection
4 unit for detecting a mutable-to-immutable conversion
5 instruction to convert a mutable string variable in which a
6 process of appending a character string is allowed, into an
7 immutable string variable in which a process of appending a
8 character string is not allowed;
9 an immutable-to-mutable conversion instruction
10 detection unit for detecting an immutable-to-mutable
11 conversion instruction to convert the immutable string
12 variable into the mutable string variable; and
13 an instruction elimination unit for eliminating the
14 immutable-to-mutable conversion instruction and for causing
15 the mutable string variable as a source variable of the
16 mutable-to-immutable conversion instruction, to be used as
17 the mutable string variable obtained from conversion by the
18 immutable-to-mutable conversion instruction after the
19 immutable-to-mutable conversion instruction, if an
20 instruction to be executed between the mutable-to-immutable
21 conversion instruction and the immutable-to-mutable
22 conversion instruction does not modify a character string
23 stored in the mutable string variable as the source variable

1 of the mutable-to-immutable conversion instruction and if an
2 instruction to be executed between the immutable-to-mutable
3 conversion instruction and use of the mutable string
4 variable obtained from the conversion by the
5 immutable-to-mutable conversion instruction does not modify
6 any of the mutable string variable as the source variable of
7 the mutable-to-immutable conversion instruction and the
8 mutable string variable obtained from the conversion by the
9 immutable-to-mutable conversion instruction.

10 14. A recording medium having any a compiler program
11 according to claim 11 recorded thereon.

12 15. A computer program product comprising a computer usable
13 medium having computer readable program code means embodied
14 therein for causing a compiler device, the computer readable
15 program code means in said computer program product
16 comprising computer readable program code means for causing
17 a computer to effect the functions of claim 1.

18 16. A computer program product comprising a computer usable
19 medium having computer readable program code means embodied
20 therein for causing a compiler device, the computer readable

1 program code means in said computer program product
2 comprising computer readable program code means for causing
3 a computer to effect the functions of claim 4.

4 17. A computer program product comprising a computer usable
5 medium having computer readable program code means embodied
6 therein for causing a compiler device, the computer readable
7 program code means in said computer program product
8 comprising computer readable program code means for causing
9 a computer to effect the functions of claim 5.

10 18. A computer program product comprising a computer usable
11 medium having computer readable program code means embodied
12 therein for causing a compiler device, the computer readable
13 program code means in said computer program product
14 comprising computer readable program code means for causing
15 a computer to effect the functions of claim 11.

16 19. A method for optimizing a program which manipulates a
17 character string, the method comprising:
18 detecting an append instruction to append a character
19 string to a string variable for storing a character string,
20 in the program;

1 generating, as a substitute for each of a plurality of
2 the append instructions detected by the append instruction
3 detection unit, a store code for storing data of an
4 appendant character string to be appended to the string
5 variable by the append instruction into a buffer, the
6 plurality of append instructions appending the character
7 strings to the same string variable; and

8 generating an append code for appending a plurality of
9 the appendant character strings to the string variable, at a
10 position to be executed before an instruction to refer to
11 the string variable in the program.

12 20. A method according to claim 19, further comprising:
13 detecting a reference instruction which first refers to
14 the string variable after the character strings have been
15 appended to the string variable by the plurality of append
16 instructions, wherein the append code generation unit
17 generates the append code at a position to be executed after
18 the store codes and before the reference instruction.

19 21. An article of manufacture comprising a computer usable
20 medium having computer readable program code means embodied
21 therein for causing optimization of a program which

1 manipulates a character string, the computer readable
2 program code means in said article of manufacture comprising
3 computer readable program code means for causing a computer
4 to effect the steps of claim 19.

5 22. A program storage device readable by machine, tangibly
6 embodying a program of instructions executable by the
7 machine to perform method steps for optimizing a program
8 which manipulates a character string, said method steps
9 comprising the steps of claim 19.